AI Open 5 (2024) 197–207

Contents lists available at ScienceDirect

AI Open

journal homepage: www.keaipublishing.com/en/journals/ai-open

Full length article

Mining contacts from spatio-temporal trajectories

Adikarige Randil Sanjeewa Madanayake *, Kyungmi Lee, Ickjai Lee

Information Technology Academy, College of Science & Engineering, James Cook University, Cairns, QLD 4811, Australia

ARTICLE INFO

Keywords:
Contact mining
Spatio-temporal trajectories
Data mining
Movement analysis

ABSTRACT

Contact mining is discovering objects in close proximity in their movements in order to reveal possible interactions, infections, collisions or contacts. This process can be significantly beneficial in a spread of an infectious disease situation to identify potential victims from a known infected human or animal, especially when the victims are asymptomatic. Movements of objects are captured by spatio-temporal trajectories represented by a series of geospatial locations and corresponding timestamps. A large amount of spatio-temporal trajectory data is being gathered by various location acquiring sensor devices by tracking movement behaviours of people, animals, vehicles and natural events. Trajectory data mining techniques have been proposed to discover useful patterns to understand the behaviours of spatio-temporal trajectories. One unexplored pattern is to identify contacts of targeted trajectory in spatio-temporal trajectories, which is defined as contact mining. The aim of this study is to investigate contact mining from spatio-temporal trajectories. The approach will be initiated by preprocessing spatio-temporal data and then by investigating a robust contact mining framework to efficiently and effectively mine contacts of a trajectory of interest from a given set of trajectories. Experimental results demonstrate the efficiency, effectiveness and scalability of our approach. In addition, parameter sensitivity analysis reveals the robustness and insensitivity of our framework.

1. Introduction

Spatio-temporal trajectory is a set of movements of an object through geographical locations over time, which can be represented by a series of geospatial coordinates (latitude, longitude) and corresponding timestamp data (Zheng, 2015). Massive amounts of data have been generated using different types of location acquiring devices. To evaluate the massiveness, if a trajectory of an object is recorded each second approximately 2.6 million spatio-temporal entries are produced for a month which will be terabytes of data for an averagely populated city.

Due to the nature of location acquiring devices, spatio-temporal trajectories suffer from a set of inherent special characteristics such as spatial uncertainties (Trajcevski, 2011) which includes irregular timestamps, over-sampled complexity, under-sampled simplicity, and measurement inaccuracy (Shamolin, 2020). Various trajectory preprocessing approaches (Enge, 1994; Shamolin, 2020) have been proposed to address these inherent special characteristics. Stay point detection (Zheng, 2015; Bermingham and Lee, 2017, 2018) also known as stop-move detection is one popular approach removing over-sampled complexities and also to add contextual information to raw spatio-temporal trajectories.

Trajectory data mining is to find interesting patterns from these large trajectories, and has been applied in many studies including traffic predictions (Ma et al., 2019), route recommendations (Qu et al., 2019), travel services (Duan et al., 2018), behaviour analysis (Yang et al., 2018), animal behaviours (Ardakani et al., 2018) and weather predictions (Miltenberger et al., 2013). Trajectory clustering and trajectory pattern mining are two widely studied areas in trajectory data mining. Trajectory clustering (Bian et al., 2018) is to group trajectories (or their segments) into similar groups to identify a subset of trajectories exhibiting similar movement patterns whilst trajectory pattern mining (Giannotti et al., 2007) is to mine frequently occurring sequential patterns or regularly occurring periodic patterns. Trajectory outlier detection is another type to identify trajectories (or their segments) substantially different from or inconsistent from others, whilst trajectory classification is to build a prediction model to classify a new trajectory into one of the pre-defined labels.

Mining contacts amongst spatio-temporal trajectories can be useful to identify potentially affected humans in an infectious disease situation such as the recent COVID pandemic outbreak. By identifying and isolating contacted individuals of a known infected individual can minimise the rapid spread of disease until a medical solution is applied. Also, tracking trajectories of humans who have been in close contact is vital to identify terrorist networks and reveal secret criminal activities. Despite the importance of contact mining, none of the existing trajectory

* Corresponding author.
E-mail addresses: adikarige.madanayake@my.jcu.edu.au (A.R.S. Madanayake), Joanne.Lee@jcu.edu.au (K. Lee), Ickjai.Lee@jcu.edu.au (I. Lee).

https://doi.org/10.1016/j.aiopen.2024.10.002
Received 25 October 2023; Received in revised form 8 July 2024; Accepted 16 October 2024
Available online 26 October 2024

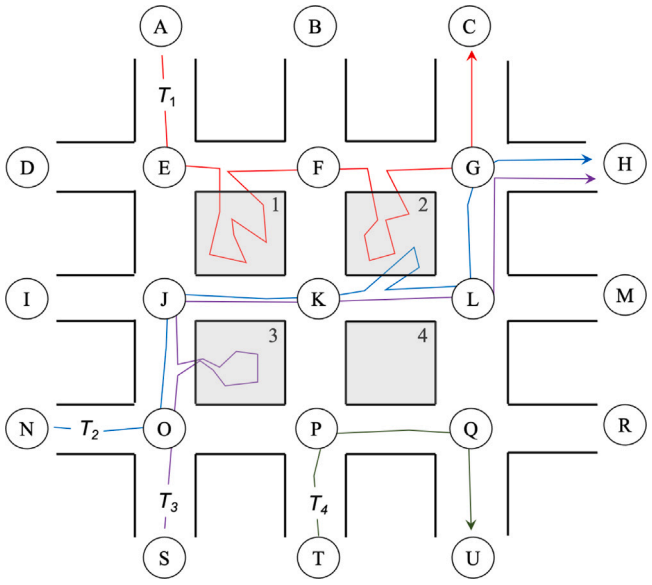


Fig. 1. An example of spatio-temporal trajectories scenario (trajectories = $\{T_1, T_2, T_3, T_4\}$).

Table 1
Movements and stops of the data shown in Fig. 1.

ID	Sequences	Stops
T_1	⟨Ⓐ,Ⓔ, Ⓕ,Ⓖ,Ⓒ⟩	Building-1 &−2
T_2	⟨Ⓝ,Ⓞ,Ⓙ,Ⓚ,Ⓛ,Ⓖ,Ⓗ⟩	Building-2
T_3	⟨Ⓢ, Ⓞ,Ⓙ,Ⓚ,Ⓛ,Ⓖ,Ⓗ⟩	Building-3
T_4	⟨Ⓣ, Ⓟ, Ⓠ, Ⓤ⟩	

preprocessing or data mining approaches is designed to mine contacts from spatio-temporal trajectories.

Fig. 1 displays a simple scenario of spatio-temporal trajectories in a given study area illustrating a gap in the literature. There are four trajectories = $\{T_1, T_2, T_3, T_4\}$ shown and their movement sequences and stops are shown in Table 1. For instance, a trajectory T_1 is moving from Ⓐ-Ⓔ-Ⓕ-Ⓖ to Ⓒ while stopping in Building-1 and Building-2. Given this dataset, trajectory clustering will group trajectories T_2 and T_3 into a cluster as they exhibit similar movements and behaviours, or sub-trajectory clustering will identify segments of trajectories (Ⓞ,Ⓙ,Ⓚ,Ⓛ,Ⓖ,Ⓗ) as a cluster. With sequential pattern mining, a sequence (Ⓞ,Ⓙ,Ⓚ,Ⓛ,Ⓖ,Ⓗ) of movements can be detected as a frequently occurring sequence whilst the trajectory T_4 can be identified as an outlier as it exhibits a different movement behaviour from other trajectories. Trajectory classification could predict the next movement based on past movements, for instance a trajectory coming from Ⓛ-Ⓖ, it is predicted to move to Ⓗ. Stay point detection is able to identify stops such as T_1 in Building-1 and Building-2, T_2 in Building-2, and T_3 in Building 3, but it is not designed to find potential contacts among moving objects. In Fig. 1, there are possible contacts between T_1 and T_2 as they stop in the same building, Building-2, also between T_2 and T_3 as both share similar movements. Unfortunately, none of the existing data mining approaches address contact mining and all fail to detect potential contacts from spatio-temporal trajectories.

This study proposes a multi-step contact mining framework that initially pre-processes raw spatio-temporal trajectories to overcome the special characteristics of spatio-temporal trajectories, then explores a hierarchical space indexing approach to efficiently and effectively detect contacts. To the best of our knowledge, it is the first attempt to investigate contact mining from massive spatio-temporal trajectories. As there does not exist ground-truth datasets, this study initially generates ground-truth contacts through the brute-force approach, and uses it as a baseline to assess performance.

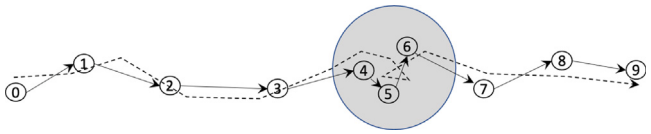


Fig. 2. An example of spatio-temporal trajectory where an actual movement trajectory is represented by a dashed line whilst a corresponding recorded trajectory is in a set of arrowed lines ($T = \{(x_0, y_0, t_0), (x_1, y_1, t_1), \ldots, (x_9, y_9, t_9)\}$. A stop is represented by a shaded area where trajectory nodes n_4, n_5, n_6 lie.

The main contributions of this study are as follows:

- formulate contact data mining;
- propose a flexible contact data mining framework for massive spatio-temporal trajectories;
- propose efficient and effective algorithmic solutions for contact data mining;
- provide various experimental results to demonstrate the performance of the proposed approach.

The rest of paper is structured as follows. Section 2 provides preliminaries whilst Section 3 reviews relevant studies identifying the literature gap. Section 4 defines the definitions and illustrates the proposed framework for contact mining. Section 5 exhibits the experimental results and presents major findings. Section 6 draws conclusive remarks and suggests possible future directions.

2. Preliminaries

A spatio-temporal trajectory (T_a) in a given spatio-temporal trajectory database $\mathcal{T} = [T_a, T_b, \ldots, T_n]$ is a list of trajectory nodes representing longitude, latitude and corresponding timestamp, denoted by $T_a = \{(x_{a1}, y_{a1}, t_{a1}), (x_{a2}, y_{a2}, t_{a2}), \ldots, (x_{an}, y_{an}, t_{an})$, where $x_{ai}, y_{ai} \in \mathbb{R}^2$ and $t_{ai} \in \mathbb{R}^+$ for $i = \{1, 2, \ldots, n\}$ and $t_{a1} < t_{a2} < \cdots < t_{an}$.

Fig. 2 displays an example of recorded spatio-temporal trajectory (line) with its actual movement (dashed line). Even though the recorded trajectory (discrete) does not exactly match with the actual movement (continuous), it captures the general movement of an entity and is a useful and solid resource for trajectory data mining. A shaded area represents a potential stop where trajectory nodes n_4, n_5, n_6 stay. As stops are places where trajectory nodes are clustered, and objects exhibit interesting behaviours, they could be potential candidates for contact mining. We explore the effect of stops in contact mining in Section 5.

3. Literature review

3.1. Spatio-temporal trajectory data mining

Trajectory data mining is the process of discovering interesting patterns such as clusters, regions-of-interest, anomalies, patterns and correlations within large trajectory datasets (Mazimpaka, 2016). Relevant main trajectory data mining approaches are covered in this subsection.

3.1.1. Stay point detection (stop-move detection)

Stop-move detection (Zheng, 2015; Bermingham and Lee, 2017, 2018) is the process of discovering and labelling entries in a trajectory that has a movement less than a specified value as stops and labelling the rest of the entries between stops as a movement. Stops and Moves of Trajectories (SMoT) (Alvares et al., 2007) was a foundational and popular algorithm used to detect stops and moves. This method was extended to several other algorithms such as SMoT+ (Moreno et al., 2014). The major drawback of these foundational algorithms is requiring parameters such as the minimum time duration which

may cause the algorithm to miss important stops. Clustering Based SMoT (CB-SMoT) (Palma et al., 2008) is a density-based algorithm, based on DBSCAN (Ester et al., 1996) which handles spatial temporal data. Density-based clustering overcomes issues such as defining the k value and initial clustering centre in partitioning-based approaches. As stops indicate places where trajectory nodes are clustered, these are considered as one of the baselines we implement for a benchmarking test in this study.

3.1.2. Trajectory clustering

Trajectory clustering is an unsupervised learning method which categorises spatio-temporal trajectory datasets into groups or clusters by identifying similarities in the same trajectories in a cluster from dissimilarities of other trajectory clusters (Bian et al., 2018). This is useful for discovering information such as object motion prediction, traffic monitoring, activity understanding, abnormal detection, and weather forecasting (Bian et al., 2018).

There are several trajectory clustering algorithms that can be categorised into partitioning-based, density-based, hierarchical-based and model-based (Bian et al., 2018). Partitioning-based algorithms are more popular since they are relatively simple and have the ability to handle large datasets. On the other hand, they have drawbacks such as the requirement of predefining the number of clusters and the impact of outliers on clustering. Although density-based clustering algorithms overcome these issues, they have their own challenges such as the requirement of pre-defined parameters and their effects in results. Also, they do not work well with high dimensional data and clusters with varying densities. Hierarchical-based algorithms overcome these issues by considering more attributes at each level, but they cost more time in computation. Model-based clustering computes internal relationships by analysing a similar matrices and hence is more efficient in processing data together. Even though trajectory clustering itself is not able to detect contacts as they can occur in non-clusters, it is worth examining the effect of clustering in contact mining as it identifies spatio-temporal aggregations where contacts could occur. Same as with the stop node detection, we implement various clustering approaches to examine the effect of clustering in contact mining in this study.

3.1.3. Trajectory classification

Trajectory classification is a supervised learning technique which classifies trajectories into already labelled pre-defined classes built using training data. This is useful when there exists pre-defined labels, and a prediction is required based on the existing ground-truth data. There are three types of classification: unsupervised, supervised and semi-supervised (Bian et al., 2019). With semantic labelling, trajectory classification is useful for many applications, such as trip recommendations, sharing life experiences, hurricane prediction and security alert triggers and context-aware computing (Patel et al., 2012; Zheng, 2015). Unsupervised and semi-supervised classification could be used as a step for contact mining, but the lack of ground-truth data is one of the main hurdles for trajectory classification to be used in contact mining.

3.1.4. Trajectory pattern mining

Trajectory pattern mining describes discoveries of significant, interesting or unexpected patterns in a movement of trajectories. Trajectory pattern mining can be categorised into several methods such as periodic/repetitive pattern mining, frequent/sequential pattern mining and moving together/group pattern mining.

Periodic or repetitive pattern mining refers to a moving object which repeatedly follows approximately the same route in different but constant time periods such as daily, monthly or annually in the same trajectory (Li et al., 2010). Identifying these kinds of behavioural patterns will be useful in predicting the future movements of these objects. This method has uncertainties since the time period affects the clustering output. The specification of a period in advance was overcome by the Periodica algorithm (Zhang et al., 2019).

Frequent or sequential pattern mining focuses on multiple moving objects that visit approximately the same place in the same order in relative time (Kopp et al., 2012; Bermingham and Lee, 2020). Frequent Spatiotemporal Sequential Pattern (FSSP) mining and Generalized Sequential Pattern (GSP) mining are some of the methods found in frequent pattern mining (Cao et al., 2005). Finding important regions from the trajectories and then applying sequential mining is a common approach to mine frequent patterns.

Group pattern mining concerns numerous moving objects staying close in a space and visiting the same locations simultaneously (Zheng, 2015). These patterns can be categorised depending on the shape and density of the group and the duration of movement of objects. There are different types of trajectories which move together in a certain time period, such as flock, convoy and swarm patterns (Zheng, 2015).

As illustrated in Fig. 1, trajectory pattern mining is designed to find frequent or regular movement patterns, and is not designed to detect contacts.

3.2. Collision detection

Detecting the intersection of geometric models is known as collision detection. Collisions should be detected when objects are static or moving. This is used in areas such as computer graphics, manufacturing, automation, robotics, computer animation, and computer simulated environments (Lin and Gottschalk, 1998; Jiménez et al., 2001; Kockara et al., 2007).

There are many collision detection algorithms available that can be categorised into two phases, such as broad phase followed by narrow phase (Kockara et al., 2009). Broad phase algorithms are initially used to identify objects that can potentially collide and exclude objects that are not colliding with certainty to optimise the speed. Then only those objects with a possibility of colliding are used to find out which objects are colliding with each other in the narrow phase. The two phases allow much more efficient collision detection than using one phase. Hubbard (1993) was the first to separate these phases and almost all the algorithms introduced follow this method.

Collision detection methods are not designed to analyse spatio-temporal trajectory data, but they are for objects in 2D or 3D context. As they are designed to handle small datasets, the scalability of these algorithms is in question. Table 2 displays a summary of related work demonstrating a gap in the literature.

3.3. Movement dynamics

To the best of our knowledge, there has been no direct study in the area of contact mining. However, recently there have been some studies to monitor asymptomatic patients (Add-Gyamfi et al., 2020; Add-Gyamfi and Zhang, 2022), to mine daily activities and movement dynamics (Yin et al., 2021; Xing et al., 2022), to investigate human activities in intra-urban networks (Liu et al., 2023a; Šveda and Madajová, 2023), and to study human movement behaviours in the COVID-19 pandemic (Majeed and Hwang, 2021; Liu et al., 2023b) from large spatio-temporal trajectories. These studies take advantage of large spatio-temporal trajectories available and mature trajectory data mining technologies to find human movement dynamics. These studies in nature focus on stay points and stops for Points of Interest (PoIs) but fail to address contacts.

Add-Gyamfi et al. (2020) investigated the movements of asymptomatic patients by examining spatio-temporal trajectories to infer their spatially and temporally bounded activities. They derived PoIs and then identified stay places where those patients visit and stay. This work has been further expanded (Add-Gyamfi et al., 2020) to continuously monitor asymptomatic patients. These approaches are similar to the stay point detection from spatio-temporal trajectories discussed in Section 3.1.1 and not designed to mine potential contacts.

Table 2
Comparison of the literature review.

Technique	Patterns	ST-Data	Contact
Stop detection	Stops/Stay points	Yes	No
Clustering	Similar groups	Yes	No
Classification	Prediction model	Yes	No
Pattern mining	Sequential/periodic	Yes	No
Traj monitoring	Stay points/PoIs	Yes	No
Collision	None	Yes	No

Liu et al. (2023a) constructed urban travel networks to understand spatial travel patterns and human activities by examining fine-scale urban travel flows and the macroscopic characteristics of the urban travel networks. Šveda and Madajová (2023) studied the distance decay of intra-urban trips in order to capture the localisation patterns of mobile phone users. These studies investigate human activities in urban networks, but they are not for mining potential contacts and interactions between humans.

Several studies expanded trajectory pattern mining discussed in Section 3.1.4 to monitor daily activity chains, to devise privacy protection techniques, and to identify connections between movement flows and land uses. Yin et al. (2021) investigated daily activity chains from spatio-temporal trajectories where they first identified stay points and then mined activity chains. This work leveraged the advances in stay point detection and trajectory pattern mining is discussed in Section 3. Xing et al. (2022) investigated a novel representation of movement dynamics within an urban area called, flow trace. This study uncovers the dynamic connections between flows and land uses, but fails to detect potential contacts.

Majeed and Hwang (2021) explored privacy protection techniques in the COVID-19 pandemic whilst Liu et al. (2023b) investigated human mobility resilience to the COVID-19 pandemic using spatio-temporal trajectories. However, these are not designed to mine potential contacts.

4. Contact data mining

4.1. Definitions

Given a set $\mathbb{T} = \{T_a, T_b, \ldots, T_n\}$ of trajectories, and let T_a be a trajectory of interest.

Definition 1 (*Spatial s-neighborhood*)**.** The spatial s-neighborhood of a trajectory node $n \in T_a$ for a given trajectory $T_i \in (\mathbb{T} \backslash T_a)$, denoted by $N_s^{T_i}(n)$, is defined by $N_s^{T_i}(n) = \{n_j \in T_i \mid dist(n_j, n) \leq s\}$, where $dist(.,.)$ is a distance function, but it is the Euclidean distance by default in this paper.

Definition 2 (*Temporal t-neighborhood*)**.** The temporal t-neighborhood of a trajectory node $n \in T_a$ for a given trajectory $T_i \in (\mathbb{T} \backslash T_a)$, denoted by $N_t^{T_i}(n)$, is defined by $N_t^{T_i}(n) = \{n_j \in T_i \mid diff(n_j, n) \leq t\}$, where $diff(.,.)$ is a time difference function that measures the difference between the two timestamps.

Definition 3 (*Spatio-temporal st-neighborhood*)**.** The spatio-temporal st-neighborhood of a trajectory node $n \in T_a$ for a given trajectory $T_i \in (\mathbb{T} \backslash T_a)$, denoted by $N_{st}^{T_i}(n)$, satisfies both Definitions 1 and 2.

Definition 4 (*Contact Duration d-neighborhood*)**.** Let $\mathbb{N}$ be a set $\{n_i, n_{i+1}, \ldots, n_{i+k}\}$ (where $i, k \in \mathbb{R}^+$) of consecutive nodes in a trajectory $T_i \in (\mathbb{T} \backslash T_a)$. The contact duration d-neighborhood of a trajectory node $n \in T_a$ for a given trajectory T_i, denoted by $N_d^{T_i}(n)$, is defined by $N_d^{T_i}(n) = \{\mathbb{N} \mid diff(n_i, n_{i+k}) \leq d\}$.

Definition 5 (*Contact Detection*)**.** A trajectory $T_i \in (\mathbb{T} \backslash T_a)$ is *contact detectable* by T_a *iff* $N_d^{T_i}(n)$ for a given d for a node $n \in T_a$ is not $\emptyset$.

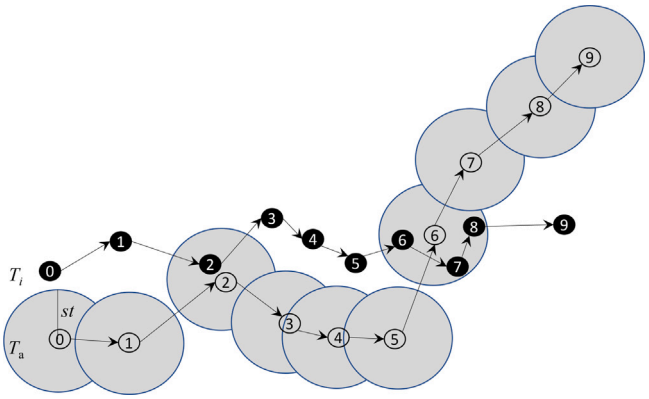


Fig. 3. Two trajectories: A trajectory $T_a = \{na_0, na_1, \ldots, na_9\}$ of interest and $T_i = \{ni_0, ni_1, \ldots, ni_9\}$.

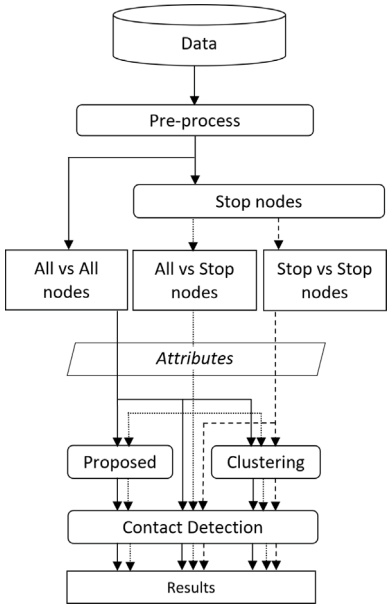


Fig. 4. Proposed framework for experimental setup of contact mining from spatio-temporal trajectories.

Definition 6 (*Contact Mining From Spatio-temporal Trajectories*)**.** For a given trajectory of interest (T_a), contact mining from a set $\mathbb{T} = \{T_a, T_b, \ldots, T_n\}$ of spatio-temporal trajectories is to find all contact detectable trajectories from $\mathbb{T} \backslash T_a$ (Definition 5).

Fig. 3 displays two trajectories: a trajectory $T_a = \{na_0, na_1, \ldots, na_9\}$ of interest and $T_i = \{ni_0, ni_1, \ldots, ni_9\}$. Shaded spheres represent spatio-temporal st-neighborhood of each node in T_a, $N_{st}^{T_i}(na)$. Please note that $N_{st}^{T_i}(na_2)$ and $N_{st}^{T_i}(na_6)$ are not $\emptyset$. In particular, for a given d-duration where $d \in \mathbb{R}^+$, $N_{st}^{T_i}(na_2)$ returns only one node ni_2 whilst $N_{st}^{T_i}(na_6)$ returns (ni_6, ni_7, ni_8). The trajectory T_i is contact detectable by T_a in this particular example when it meets two conditions: (1) $diff(ni_6, ni_8) \leq d$, and (2) nodes ni_6, ni_7, ni_8 satisfy Definition 3.

4.2. Proposed framework

A multi-step hierarchical contact mining framework is proposed to identify contacts of a trajectory of interest, as shown in Fig. 4. Please note that in our proposed framework, we compare the performance of our proposed method against several baselines. As discussed in Section 3, clustering and stop node detection are two data mining approaches that could be used in contact mining, we transform datasets

with stops and implement several clustering algorithms for comparison. Initially, raw data is gathered and cleaned through a data preprocessing step in order to remove the inaccuracies and irregularities in the raw trajectory data to ensure consistency for contact mining. Subsequently, a stop node detection process is implemented to distinguish stop nodes from movement nodes.

Thereafter, three types of datasets are used to carry out the rest of the process. The first dataset is with all nodes in the trajectory data without any trajectory simplification. That is, all nodes in the trajectory of interest and all nodes in other trajectories are being used. Others are with trajectory simplification using stop nodes detection. The second one is with all nodes in the trajectory of interest and stop nodes in other trajectories. The third one is with stop nodes in the trajectory of interest and stop nodes in all other trajectories. The second and third datasets are simpler than the first dataset, and these are to investigate the effect of stop node detection in contact mining. Once datasets are generated, a brute-force contact mining approach is applied to produce ground-truth contacts. This is a time-demanding process that requires exhaustively going through datasets to generate ground-truth contacts. The output will be used as a benchmark baseline for other approaches. A set of various clustering approaches is used to explore any efficiency or effectiveness improvements in contact mining.

4.2.1. Data gathering

The data contains a unique identification number for each trajectory, and each node of trajectories is represented by latitude, longitude and corresponding timestamp. The contact mining framework is tested in stages with three different types of spatio-temporal data. In the initial stage, small synthetic datasets are created to test the accuracy of the program in diverse scenarios in the framework. Later, a series of large synthetic datasets ranging from 100 thousand to 10 million nodes with varying number of trajectories are created and used to evaluate the scalability of our approach. Synthetic datasets used in the paper have been generated through pseudo-random generators in order to simulate random spatio-temporal behaviours (Liu et al., 2024; Bhattacharjee and Das, 2022).

Subsequently, a personal movement real dataset is collected to capture a real-world scenario and used for accuracy analysis as well as for parameter sensitivity analysis. In addition, a real-world dataset (Zheng et al., 2010) has been downloaded and processed for accuracy analysis.

4.2.2. Data preprocessing

Complex and large spatio-temporal trajectories are uncertain, nuanced and irregular. They have to be represented in a simplified format by reducing complexities, inaccuracies, inconsistencies, uncertainties, and irregularities while preserving the underlying structures and general movement patterns for effective and efficient data mining. Several data preprocessing steps have been utilised in this paper. First, measurement inaccuracies and spatial uncertainties (due to low quality antennas, surrounding barriers, or weather conditions), have been handled by removing inaccurate and incomplete data. Second, inconsistencies in data coming from different devices have been managed to convert them into a consistent format. Table 3 shows an example of converting different types of date-time into a consistent timestamp format. Third, raw datasets are nuanced and irregular with respect to timestamps, which requires a preprocessing step in order to remove any irregularities and inconsistencies in the data. For instance, timestamps could be irregular as $diff(n_i, n_{i+1}) \neq diff(n_j, n_{j+1})$ for $n_i, n_j \in T$. This preprocessing is to make each trajectory regular to be $diff(n_i, n_{i+1}) == diff(n_j, n_{j+1}) \; \forall \; i \neq j$.

4.2.3. Stay node detection (stop-move detection)

Stop node detection could be a useful process for contact mining as it identifies stopping nodes where objects stay for a specific duration in a specific spatio-temporal range, resulting in physical contacts. There have been many different stop node detection approaches that have been proposed. This paper implements CB-SMoT (Palma et al., 2008) approach to find stop nodes.

Table 3
Transformation of date-time into a consistent format.

Date	Time	Timestamp
1st August 2020	1:10:01 pm	20200801:131001
2/8/2020	13:10:02	20200802:131002
3/8/2020	1:10:03	20200801:131003

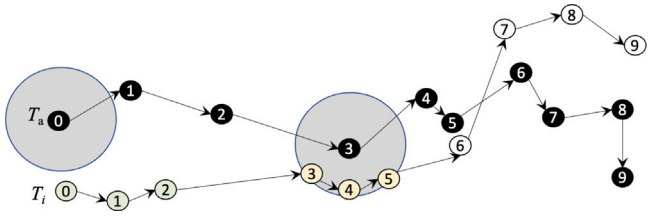


Fig. 5. Two trajectories: A trajectory $T_a = \{na_0, na_1, \ldots, na_9\}$ of interest and $T_i = \{ni_0, ni_1, \ldots, ni_9\}$. Two enclosing shaded hypersphere around na_0 and $na_3 \in T_a$ represent the corresponding d-neighborhood of those nodes whilst two clusters $[(ni_0, ni_1, ni_2), (ni_3, ni_4, ni_5)]$ in T_i are shaded with different colours.

4.2.4. User-provided attributes

Our framework requires several user-specified attributes to fine-tune our algorithm to be flexible and applicable to various scenarios. It receives a spatial neighborhood threshold s, a temporal neighborhood threshold t, and a duration d in order to identify contacts. A set of various experiments has been conducted in Section 5 for sensitivity analysis of these parameter values.

4.2.5. Contact mining

Once datasets are pre-processed and a set of user-provided attributes are given, contact mining could be applied. In our paper, three approaches are implemented.

First, a naive brute-force approach is performed to identify ground-truth contacts. In this approach, all nodes in a given trajectory-of-interest is compared to all other nodes in other trajectories. That is, given a set $\mathbb{T} = \{T_1, T_2, \ldots, T_n\}$ of spatio-temporal trajectories and a trajectory of interest $T_a \notin \mathbb{T}$, this approach requires $|T_a| \times |T_1| \times |T_2| \times \cdots \times |T_n|$ operations. In other words, this exhaustive search strategy (Nievergelt, 2000) comprehensively explores every single case to find ground-truth contacts. This guarantees that all correct contacts are found by listing all the possible candidate contacts. This is time-consuming, but is necessary to identify ground-truth contacts for benchmarking purposes.

Second, a clustering-based approach can be used to prune the search space and to improve the efficiency as clustering identifies aggregations/concentrations where contacts could occur. Clustering is to group spatially and temporally similar points into the same cluster, thus it is a strong candidate for narrowing down the search space. There have been many different clustering approaches in the literature, but the most popular and widely used spatio-temporal clustering approaches have been utilised in this paper including DBSCAN (Ester et al., 1996), OPTICS (Ankerst et al., 1999), k-Means (Hartigan and Wong, 1979), and BIRCH (Zhang et al., 1996). As the number of clusters is much smaller than the number of entities, this approach improves the efficiency at the expense of effectiveness (producing false positives and false negatives).

For given two trajectories shown in Fig. 5, a trajectory $T_a = \{na_0, na_1, \ldots, na_9\}$ of interest and $T_i = \{ni_0, ni_1, \ldots, ni_9\}$, the naive brute-force approach exhaustively searches for a contact. That is, each node in T_i is checked against every node in T_a to see if it is a contact. In this particular example, let us assume there are two contacts identified through the brute-force approach: (na_3 and ni_4) and (na_5 and ni_6) satisfying st-neighborhood (Definition 3) and d-neighborhood (Definition 4). On the other hand, the clustering approach first finds clusters from T_i and checks them individually to the nearest node in

Algorithm 1 Find_Contacts

Input:
$dsTOI$: Trajectory Of Interest in the given dataset;
$dsOT$: Other Trajectories in the given dataset;
$dsAttributes$:User-provided attributes for the dataset;
Output:
$dsContactsFound$: Contacts found in the given dataset;

function FIND_CONTACTS($dsTOI, dsOT, dsAttributes$)
  Create an empty list $dsContactsFound$;
  Assign $dsAtributes$ to (s, t, d); //neighborhood values
  $nDID = s$ and t neighborhood in degrees;
  for each node in $dsTOI$ **do**
    Compute bounding cube of the node using $nDID$;
    node++;
  end for
  Arrange $dsTOI$ in an ascending order;
  while not end of $dsTOI$ **do**
    Assign $False$ to $IsFoundContact$;
    while not end of $dsOT$ **do**
      if $dsOT$ $node$ within bin of $dsTOI$ **then**
        Call $Find\_Nodes()$ to $IsFoundContact$;
        if $IsFoundContact = True$ **then**
          exit While;
        end if
      end if
    end while
    if $IsFoundContact == True$ **then**
      Assign $dsOT[id]$ to $OTid$;
      Append $OTid$ to $dsContactsFound$;
      while $dsOT[id] == OTid$ **do**
      end while
    end if
  end while
  return $dsContactsFound$;
end function

Algorithm 2 Find_Nodes

Input:
$dsTOI$: Trajectory Of Interest in the given dataset;
$dsOT$: Other Trajectories in the given dataset;
$dsAttributes$:User-provided attributes for the dataset;
Output:
$IsContactFound$: Contact found or not;

function FIND_NODES($dsTOI, dsOT, dsAttributes$)
  Assign $dsAtributes$ to (s, t, d); //neighborhood values
  Assign $False$ to $IsContactFound$;
  Assign Ø to $ContactNodesFound$;
  Assign $dsOT[id]$ to $OTid$;
  while $dsOT[id] == OTid$ **do**
    Compute distance between nodes of
    $dsTOI$ and $dsOT$ to $Distance$;
    if $Distance > s$ **then**
      exit While;
    end if
    Compute time variance between nodes of
    $dsTOI$ and $dsOT$ to $Delay$;
    if $Delay > t$ **then**
      exit While;
    end if
    Add 1 to $NodesFound$;
    if $NodesFound == 1$ **then**
      Assign $dsOT[timestamp]$ to $StartTime$;
    else
      Compute time variance between
      $StartTime$ and $dsOT[time]$ to $Duration$;
      if $Duration \geq d$ **then**
        Assign $True$ to $IsContactFound$;
        exit While;
      end if
    end if
  end while
  return $IsContactFound$;
end function

T_a to see if it is a contact. This clustering based approach reduces the search space, but it misses potential true contacts such as na_5 and ni_6.

Finally, our approach Minimum Bounding Cube (MBC) is used to identify contacts. This will calculate the actual distance obtained from attributes to create a minimum bounding cube from each node of the trajectory of interest. Then, it will check whether each node of other trajectories falls within these cubes to meet the st-neighborhood and d-neighborhood. Our MBC approach is based on the following two algorithms. Algorithm 1 is the main algorithm to find all the contacts, whilst Algorithm 2, called by Algorithm 1, is used to find each individual node. Algorithm 1 first takes three variables as input, and they include: a trajectory of interest in a given dataset (denoted by $dsTOI$), other trajectories in the given dataset ($dsOT$,) and a set of user-provided neighborhood attributes for the dataset ($dsAttributes$), and outputs a list of contacts found with the given set of input data. First, the algorithm computes spatio-temporal neighborhood in degrees in order to create neighbouring cubes for geo-locations recorded in latitude and longitude. For each node in $dsTOI$, Algorithm 1 iterates each trajectory in $dsOT$ to see if there are contacts for each node in $dsTOI$ by calling Algorithm 2, which returns a boolean value indicating whether or not a contact is found between $dsTOI$ and each trajectory in $dsOT$.

Given a set $\mathbb{T} = \{T_a, T_b, \ldots, T_n\}$ of trajectories, and let T_a be a ToI, m be the number of nodes in T_a ($|T_a|$), and M be the total number of nodes in $\mathbb{T} \backslash T_a$. Algorithm 1 calls Algorithm 2 for m times which requires $O(\log M)$ as it is a region query (Galán, 2019). Thus the time complexity of proposed algorithm requires $O(m \log M)$.

5. Experimental results

All our experiments are carried out on a machine with an Intel(R) Core(TM) i7-8750H @2.20 GHz processor and 20 GB unallocated memory. All programs are implemented in Python.

Initially, a set of real-world trajectory datasets (Zheng et al., 2010) ranging from 50,000 to 500,000 nodes (denoted by $\mathbb{T}1$) has been downloaded and processed with a set of user-defined attributes to identify contacts. This dataset is used to measure the accuracy of baselines. Also, this dataset with a range of different attribute values (s-neighborhood, t-neighborhood, d-neighborhood) has been used to evaluate the adaptability and flexibility of our proposed framework. Subsequently, a much larger set of synthetic datasets has been generated to evaluate the scalability of our framework. Please note that, the brute-force approach and other baselines are not scalable, and it is impractical to test their scalability with larger datasets. Also note that the stop node detection and various clustering based approaches have been implemented to evaluate their suitability and performance in various settings.

In all our experiments, several runs of each clustering approach have been conducted to find the best clustering result for each clustering approach, thus hyperparameters of those clustering algorithms vary with the datasets used. Those hyperparameters of our approach remain the same for Table 4, Table 5, Table 6, Fig. 6, Fig. 7 and Fig. 8: 2 meters for s-neighborhood; 1 day for t-neighborhood; and 5s for d-neighborhood for consistency in experiments. A parameter sensitivity analysis (with various hyperparameters in Table 7, Table 8 and Fig. 9)

Table 4
Accuracy performance with all nodes in dataset $\mathbb{T}1$ (without stop node detection).

Clustering\Nodes	50k	100k	250k	500k
Brute-force	5	8	21	35
DBSCAN	4	7	18	31
OPTICS	4	7	17	30
k-Means	3	5	15	29
BIRCH	4	7	18	30
MBC	5	8	21	35

Table 5
Accuracy performance with stop nodes in dataset $\mathbb{T}1$ (with stop node detection).

Clustering\Nodes	50k	100k	250k	500k
Brute-force	1	3	6	8
DBSCAN	1	3	6	8
OPTICS	1	3	6	8
k-Means	1	3	6	8
BIRCH	1	3	6	8
MBC	5	8	21	35

Table 6
Efficiency performance of the brute-force approach with and without stop node detection for dataset $\mathbb{T}1$ (in seconds).

Stop node\Nodes	50k	100k	250k	500k
Without detection	16,911	33,694	85,492	170,024
With detection	8,455	17,520	38,471	79,911

has been conducted in Table 7 to demonstrate the insensitivity of our approach against hyperparameters.

5.1. Accuracy analysis with/without stop node detection

According to the experiments carried out to identify contacts, using stop nodes in trajectories was found to be more efficient than using all nodes in trajectories in all approaches used. This is simply due to the number of stop nodes for a given trajectory $T_i \in (\mathbb{T}\backslash T_a)$ is less than or equal to the number of all nodes for T_i, in most cases much lesser which compensates the extra stop node detection time. Table 6 demonstrates the efficiency performance of the brute-force approach with and without stop node detection for dataset $\mathbb{T}1$. However, it was observed that the accuracy of identifying contacts deteriorated when the stop node detection was used as it missed some true positives. That is, there is a trade-off between accuracy and efficiency with the use of stop nodes in contact mining. Identifying contacts using stop nodes is useful when efficiency is more important than accuracy. Table 4 displays accuracy performance identifying contacts with all nodes in the dataset (without stop node detection), whilst Table 5 shows that with stop nodes. As shown in Table 4, the brute-force approach identifies 5 ground-truth contacts for the dataset with 50k nodes, 8 contacts with 100k, 21 contacts with 250k and 35 contacts with 500k with all nodes in the dataset (without the stop node detection). However, as shown in Table 5 the brute-force approach detects a much smaller number of contacts with the stop node detection. This demonstrates that the contact mining with the stop node detection misses some true positives at the gain of efficiency. Tables 4 and 5 also show that the performance of contact mining with various clustering approaches deteriorates regardless of the clustering approaches used. However, it is evident that the proposed MBC approach is able to detect all the ground-truth contacts for all datasets used producing the same result as the brute-force approach with the real-world dataset $\mathbb{T}1$.

5.2. Efficiency analysis

Fig. 6 displays processing times in seconds for $\mathbb{T}1$ with the brute-force (base) approach, along with various clustering approaches and our proposed method MBC. As explained in Section 4.2.5, the main procedure of MBC is Algorithm 1, which calls Algorithm 2 for mining contacts, the processing time of MBC in Fig. 6 includes both times of Algorithm 1 and Algorithm 2.

Table 7
Parameter sensitivity analysis with varying number of trajectories and a fixed number ($\|n\| = 1000$) of nodes per trajectory.

NBHD	(2m-1d-5s)					(2m-1h-5s)					(3m-1h-5s)					(2m-1h-15s)				
APR\TRAJ	10	20	30	40	50	10	20	30	40	50	10	20	30	40	50	10	20	30	40	50
AA-BB	**3**	**5**	**7**	**10**	**14**	**1**	**2**	**4**	**4**	**6**	**1**	**2**	**6**	**7**	**9**	**1**	**1**	**1**	**1**	**1**
AA-CD	1	0	1	1	2	0	0	1	1	2	0	1	2	2	3	0	0	0	0	0
AA-CO	0	0	1	1	2	0	0	1	1	2	0	1	2	2	3	0	0	0	0	0
AA-CK	1	3	5	5	8	1	2	4	4	4	1	2	5	5	6	0	1	1	1	1
AA-CB	1	2	5	5	7	1	2	4	4	6	1	2	5	5	6	0	1	1	1	1
AA-OM	**3**	**5**	**7**	**10**	**14**	**1**	**2**	**4**	**4**	**6**	**1**	**2**	**6**	**7**	**9**	**1**	**1**	**1**	**1**	**1**
AS-BB	**2**	**4**	**6**	**9**	**13**	**1**	**2**	**4**	**4**	**6**	**1**	**2**	**5**	**6**	**8**	**1**	**1**	**1**	**1**	**1**
AS-CD	0	0	1	1	1	0	0	1	1	1	0	1	2	2	3	0	0	0	0	0
AS-CO	0	0	1	1	1	0	0	1	1	1	0	1	2	2	3	0	0	0	0	0
AS-CK	1	3	5	5	4	1	2	4	4	4	1	2	5	6	6	0	1	1	1	1
AS-CB	1	4	5	2	4	1	2	3	3	3	1	2	5	6	3	0	1	1	1	1
AS-OM	**2**	**4**	**6**	**9**	**13**	**1**	**2**	**4**	**4**	**6**	**1**	**2**	**5**	**6**	**8**	**1**	**1**	**1**	**1**	**1**
SS-BB	**1**	**2**	**3**	**5**	**8**	**1**	**1**	**2**	**2**	**4**	**1**	**2**	**4**	**4**	**6**	**1**	**1**	**1**	**1**	**1**
SS-CD	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
SS-CO	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
SS-CK	0	2	3	2	3	0	1	2	2	2	0	2	4	4	5	0	1	1	1	1
SS-CB	0	2	3	2	5	0	1	2	2	2	0	2	4	4	5	0	1	1	1	1
SS-OM	**1**	**2**	**3**	**5**	**8**	**1**	**1**	**2**	**2**	**4**	**1**	**2**	**4**	**4**	**6**	**1**	**1**	**1**	**1**	**1**

NBHD: Neighborhood (s-neighborhood, t-neighborhood, d-neighborhood); (2m-1d-5s): (s-neighborhood = 2 m, t-neighborhood = 1 day, d-neighborhood = 5 s); (2m-1h-5s): (s-neighborhood = 2 m, t-neighborhood = 1 h, d-neighborhood = 5 s); (3m-1h-5s): (s-neighborhood = 3 m, t-neighborhood = 1 h, d-neighborhood = 5 s); (2m-1h-15s): (s-neighborhood = 2 m, t-neighborhood = 1 h, d-neighborhood = 15 s); ARP: Approaches; TRAJ: number of trajectories; AA: All nodes in ToI and All nodes in other trajectories; AS: All nodes in ToI and Stops in other trajectories; SS: Stops in ToI and Stops in other trajectories; BB: Bruteforce Baseline; CD: Clustering with DBSCAN; CO: Clustering with OPTICS; CK: Clustering with k-means; CB: Clustering with BIRCH, OM: Our proposed Method.

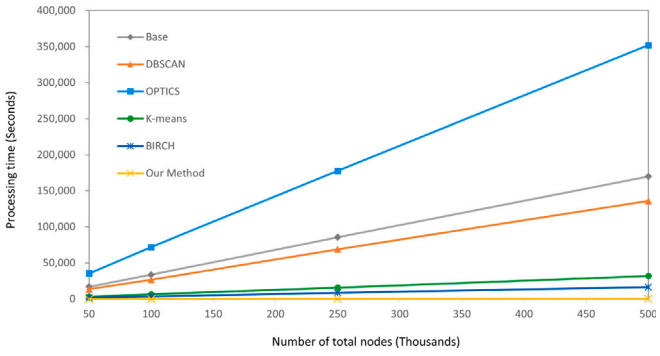


Fig. 6. Processing time in seconds to identify contacts with dataset $\mathbb{T}1$.

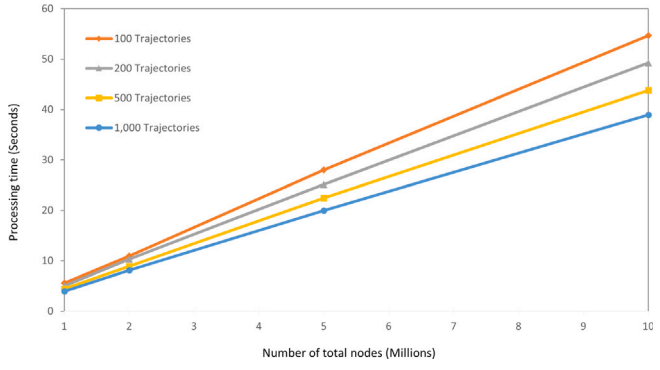


Fig. 7. Scalability analysis for each fixed number of trajectory with varying number of nodes per trajectory for dataset $\mathbb{T}2$.

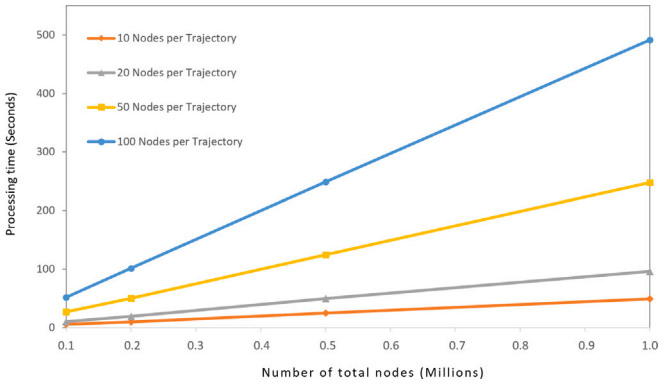


Fig. 8. Scalability analysis for each fixed number of nodes per trajectory with varying number of trajectories for dataset $\mathbb{T}2$.

Table 8
Parameter sensitivity analysis with a fixed number ($\|n\| = 25$) of trajectories and a varying number of nodes per trajectory.

NBHD	(2m, 1d, 5s)					(2m, 1h, 5s)				
APR\NODES	500	1000	1500	2000	25 000	500	1000	1500	2000	2500
AA-BB	**2**	**7**	**9**	**11**	**13**	**1**	**5**	**6**	**6**	**6**
AA-CD	1	2	5	4	4	1	2	3	3	4
AA-CO	1	2	5	4	4	1	2	3	3	4
AA-CK	1	5	8	8	9	**1**	**5**	**6**	**6**	**6**
AA-CB	1	5	9	10	10	**1**	**5**	**6**	**6**	**6**
AA-OM	**2**	**7**	**9**	**11**	**13**	**1**	**5**	**6**	**6**	**6**
AS-BB	**1**	**7**	**9**	**11**	**13**	**1**	**5**	**6**	**6**	**6**
AS-CD	0	1	2	3	4	0	1	3	3	3
AS-CO	0	1	2	3	4	0	1	3	3	3
AS-CK	1	5	8	8	9	**1**	**5**	**6**	**6**	**6**
AS-CB	1	6	7	7	6	1	4	6	6	6
AS-OM	**1**	**7**	**9**	**11**	**13**	**1**	**5**	**6**	**6**	**6**
SS-BB	**0**	**4**	**6**	**9**	**10**	**0**	**3**	**4**	**5**	**5**
SS-CD	0	1	1	1	1	0	1	1	1	1
SS-CO	0	1	1	1	1	0	1	1	1	1
SS-CK	0	3	5	7	8	**0**	**3**	**4**	**5**	**5**
SS-CB	0	3	4	6	9	**0**	**3**	**4**	**5**	**5**
SS-OM	**0**	**4**	**6**	**9**	**10**	**0**	**3**	**4**	**5**	**5**

Please note that several runs of each clustering approach have been conducted to find the best clustering result for each clustering approach, and those times taken for clustering approaches include both clustering time and contact mining time.

As you can see from Fig. 6, the processing time linearly increases as the size of nodes in trajectories grows. Interestingly with this particular dataset, contact mining with OPTICS is the least efficient, and this is because OPTICS produces the most number of clusters with this dataset, resulting in less computational gain in the subsequent contact mining process. However, the rest of clustering approaches including DBSCAN and k-means and BIRCH are more efficient than the brute-force approach. Notably, our proposed approach is the most efficient as it efficiently prunes the search space through MBC without requiring extra clustering time.

5.3. Scalability analysis

For this scalability analysis of our MBC approach, we have created a set of large synthetic datasets (denoted by $\mathbb{T}2$) with varying number of nodes and trajectories. Fig. 7 displays a scalability analysis with fixed number of trajectories having varying number of nodes per trajectory whilst Fig. 8 shows fixed number of nodes per trajectory having varying number of trajectories. The synthetic dataset $\mathbb{T}2$ has been created to include from 1 million to 10 million nodes for 100 trajectories, 200 trajectories, 500 trajectories and 1000 trajectories for the first scalability analysis, and from 10 to 100 nodes per trajectory for 0.1 million, 0.2 million, 0.5 million and 1 million total nodes for the second scalability analysis. Please note that the scale of data used in Fig. 8 is 10 times more than the one in Fig. 7.

As you can see from Fig. 7, our approach is scalable as it is able to process these trajectories with up to 10 millions nodes within a minute. That is, it displays a linear growth. Also, a similar trend illustrated by scalability graphs with varying number of nodes demonstrates the independence of our algorithm from the number of nodes. Interestingly, dense trajectories (with more number of nodes) take slightly more time than sparse trajectories as shown in Fig. 7 and this is because our approach requires the examination of more nodes for each MBC with dense trajectories. This is somehow consistent with the real-world scenario where slow moving entities (people) within a small space (such as a shopping centre) resulting in dense trajectories would require more checks for contact mining. Fig. 8 displays a linear growth in time is required as the number of trajectories increases. Similar to Fig. 7, Fig. 8 shows a similar increasing trend regardless of the number of trajectories which demonstrates the independence of our algorithm from the number of trajectories demonstrating our algorithm is scalable to the number of trajectories.

5.4. Parameter sensitivity analysis

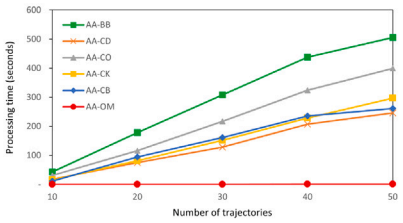
For this parameter sensitivity analysis, we have generated a set of small personally collected real-world datasets (denoted by $\mathbb{T}3$) to evaluate the sensitivity of attributes (parameters) (s-neighborhood, t-neighborhood, and d-neighborhood) used in our approach. As the identification of ground-truth contact is time consuming and other approaches using the stop node detection and clustering approaches are not scalable, the datasets used in this parameter sensitivity analysis are kept small to be computationally manageable.

Table 7 displays parameter sensitivity analysis with varying number of trajectories and a fixed number of ($\|n\| = 1000$) of nodes per trajectory. First of all, it is evident that our proposed methods (AA-OM, AS-OM and SS-OM) are able to find all ground-truth contacts for datasets with various parameter settings. That is, AA-OM finds all 3 contacts (2m-1d-5s) parameter (attribute) values for the 10 trajectory dataset, 5 contacts for the 20 trajectory dataset, 7 for the 30 trajectory dataset, 10 for the 40 trajectory dataset, and 14 contacts for the 50 trajectory dataset. The incorporation of stop node detection in contact mining does not deteriorate the performance of our proposed method as our proposed methods AS-OM and SS-OM are able to detect all ground-truth contacts in various parameter settings in this experiment. For example, the numbers of contacts identified by AA-OM are the same as those by AA-BB, and similarly the contact numbers by AS-OM are the same as those by AS-BB, and also the numbers by SS-OM are the same as those by SS-BB for all NBHD settings for all datasets in Table 7.

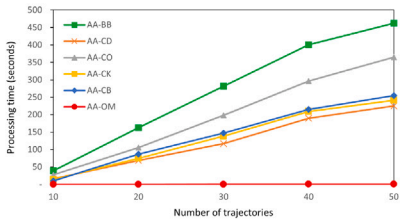
Second, the stop node detection as a preprocessing step in contact mining is likely to miss some true positives as AS-BB finds a smaller number of contacts than AA-BB. It is also consistent with SS-BB, which finds fewer contacts than AS-BB in most settings except (2m-1h-15s) where only one contact is in the dataset, so the effect could be minimal. This confirms that the use of stop node detection achieves efficiency improvements at the expense of effectiveness.

Third, clustering approaches can be integrated into contact mining for efficiency improvement. However, as shown in Table 7 all clustering approaches are underperforming, and they miss many true positive contacts. This is consistent with and without the stop node detection implemented. One interesting finding is that clustering approaches with k-means and BIRCH outperform those with DBSCAN and OPTICS.

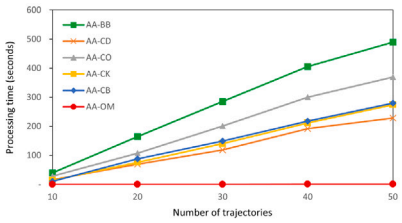
Fourth, as expected the number of ground-truth contacts decreases with an increase in t-neighborhood as the increase strengthens the contact mining requirement. For instance the NBHD setting (2m-1h-5s) detects 1:2:4:4:6 contacts whilst the (2m-1d-5s) setting finds 3:5:7:10:14 for 10:20:30:40:50 trajectories. A similar trend is observed with d-neighborhood as the increase in d-neighborhood will result in a less



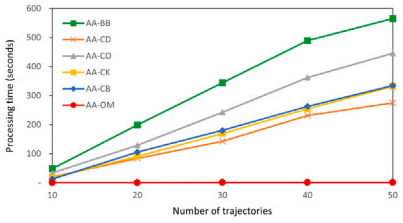
(a) Attributes: s-neighborhood=2m, t-neighborhood=1 day, d-neighborhood=5 seconds.



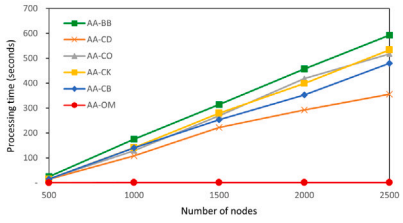
(b) Attributes: s-neighborhood=2m, t-neighborhood=1 hour, d-neighborhood=5 seconds.



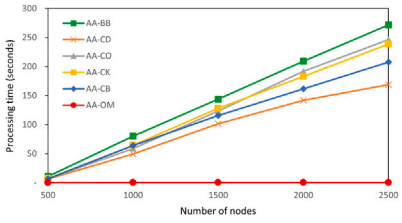
(c) Attributes: s-neighborhood=3m, t-neighborhood=1 hour, d-neighborhood=5 seconds.



(d) Attributes: s-neighborhood=2m, t-neighborhood=1 day, d-neighborhood=15 seconds.



(e) Attributes: s-neighborhood=2m, t-neighborhood=1 day, d-neighborhood=5 seconds.



(f) Attributes: s-neighborhood=2m, t-neighborhood=1 hour, d-neighborhood=5 seconds.

Fig. 9. Efficiency analysis with various parameter settings for $\mathbb{T}3$.

number of contacts. Oppositely, a rise in s-neighborhood increases the number of ground-truth contacts as it relaxes the contact data mining requirement. For instance, the NBHD setting (3m-1h-5s) detects 1:2:6:7:9 whilst (2m-1h-5s) finds 1:2:4:4:6 for 10:20:30:40:50 trajectories.

Similar trends are observed with varying nodes per trajectory, as shown in Table 8. Our proposed method detects all ground-truth contacts in all settings with and without the stop node detection. AA-OM produces the same results as AA-BB, AS-OM generates the same results as AS-BB, whilst SS-OM yields the same results as SS-BB for both NBHD settings (2m, 1d, 5s) and (2m, 1h, 5s). Also, an increase in t-neighborhood results in more number of contacts regardless of the number of nodes, which is consistent with Table 7 .

Fig. 9 displays efficiency performance with various parameter settings. Efficiency analysis with varying numbers of trajectories with different attribute settings are shown from Fig. 9(a) to Fig. 9(d) whilst those with varying number of nodes with different attribute settings are depicted in Fig. 9(e) to Fig. 9(f). In all cases, our approach (AA-OM) outperforms the baselines with different clustering approaches. Please note that the use of clustering algorithms improves efficiency at the expense of accuracy, as discussed earlier, and this is evident in Fig. 9 where clustering-based approaches all perform faster than AA-BB in all parameter settings. Please note that our approach even outpaces these efficiency-improved clustering approaches for all.

5.5. Discussion

A comprehensive set of experimental results presented in this section demonstrates the outperforming performance and favourable results of our proposed contact mining approach in various aspects against baselines used. Section 5.1 reveals that our approach is able to find all potential contacts with/without the stop node detection procedure in place (identified by the brute-force approach) for the datasets under study. This demonstrates that our approach produces more accurate and favourable results than the baselines used. Section 5.2 demonstrates the efficacy of MBC whilst Section 5.3 shows the scalability of MBC with various sizes of datasets. Efficiency and scalability analyses demonstrate the outstanding performance of our proposed method while detecting all ground-truth true positive contacts for various datasets under study. Section 5.4 demonstrates the insensitivity of MBC to parameter values consistently outperforming baselines in various parameter settings. This demonstrates the flexibility and robustness of our approach. Also, various parameter values with st-neighborhood and d-neighborhood allow our proposed system to be flexible and applicable to diverse disease transmission and contact discovery scenarios.

6. Conclusion

Contact data mining is an interesting topic as it investigates potential contacts involving interaction and communication with others. It could be used to identify suspicious interactions and meetings in criminal network analysis, to spot social interactivities in behaviour analysis, and to recognise potential interactions in epidemic disease analysis. The availability of spatio-temporal trajectories enables us to identify such contacts using data mining approaches. Despite the need for contact data mining, no previous study has been conducted in this area to the best of our knowledge.

This paper defines contact data mining, and proposes a contact data mining framework for spatio-temporal trajectories. This study further explores two popular data mining methods that could be integrated into contact data mining: stop node detection and clustering. These two methods are of particular interest as the stop node detection identifies potential stops where typically objects exhibit meaningful interactions and also clustering identifies spatio-temporal aggregations where entities show similar spatio-temporal characteristics. This paper proposes a MBC based search space pruning approach for contact data mining, which is efficient and scalable.

There are a few folds for future studies. First, we would like to explore contact mining with multiple trajectories-of-interest. Second, as evidenced in the spread of epidemic disease, multiple levels of contacts could be studied to identify the growth and trend of contacts. Third, an (semi-) or automated generation of ground-truth datasets with many contacts is to be explored. The lack of ground-truth real-world datasets complicates contact data mining, a simulation approach could be investigated to automatically generate a large number of ground-truth contacts.

CRediT authorship contribution statement

Adikarige Randil Sanjeewa Madanayake: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Kyungmi Lee:** Project administration, Supervision, Writing – review & editing. **Ickjai Lee:** Conceptualization, Project administration, Supervision, Writing – review & editing.

Declaration of competing interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed.

We further confirm that the order of authors listed in the manuscript has been approved by all of us. We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

We understand that the Corresponding Author is the sole contact for the Editorial process (including Editorial Manager and direct communications with the office). He/she is responsible for communicating with the other authors about progress, submissions of revisions and final approval of proofs.

References

Add-Gyamfi, D., Zhang, F., 2022. Mobility and trajectory-based technique for monitoring asymptomatic patients. J. Inf. Technol. Res. 15 (1), http://dx.doi.org/10.4018/JITR.2022010109.

Add-Gyamfi, D., Zhang, F., Kwanzaa Ansah, A.K., 2020. EDDAMAP: efficient data-dependent approach for monitoring asymptomatic patient. BMC Med. Inform. Decis. Mak. 20 (245), http://dx.doi.org/10.1186/s12911-020-01258-z.

Alvares, L., Bogorny, V., Kuijpers, B., Macedo, J., Moelans, B., Vaisman, A., 2007. A model for enriching trajectories with semantic geographical information. In: Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems. p. 22. http://dx.doi.org/10.1145/1341012.1341041.

Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J., 1999. OPTICS: Ordering points to identify the clustering structure. SIGMOD Rec. 28 (2), 49–60. http://dx.doi.org/10.1145/304181.304187.

Ardakani, I., Hashimoto, K., Yoda, K., 2018. Understanding animal behavior using their trajectories. pp. 3–22. http://dx.doi.org/10.1007/978-3-319-91131-1_1.

Bermingham, L., Lee, I., 2017. A framework of spatio-temporal trajectory simplification methods. Int. J. Geogr. Inf. Sci. 31 (6), 1128–1153.

Bermingham, L., Lee, I., 2018. A probabilistic stop and move classifier for noisy GPS trajectories. Data Min. Knowl. Discov. 32 (6), 1634–1662.

Bermingham, L., Lee, I., 2020. Mining distinct and contiguous sequential patterns from large vehicle trajectories. Knowl.-Based Syst. 189 (C), http://dx.doi.org/10.1016/j.knosys.2019.105076.

Bhattacharjee, K., Das, S., 2022. A search for good pseudo-random number generators: Survey and empirical studies. Comp. Sci. Rev. 45, 100471. http://dx.doi.org/10.1016/j.cosrev.2022.100471, URL: https://www.sciencedirect.com/science/article/pii/S1574013722000144.

Bian, J., Tian, D., Tang, Y., Tao, D., 2018. A survey on trajectory clustering analysis.

Bian, J., Tian, D., Tang, Y., Tao, D., 2019. Trajectory data classification: A review. ACM Trans. Intell. Syst. Technol. 10 (4), http://dx.doi.org/10.1145/3330138.

Cao, H., Mamoulis, N., Cheung, D., 2005. Mining frequent spatio-temporal sequential patterns. In: Proceedings - IEEE International Conference on Data Mining. ICDM, http://dx.doi.org/10.1109/ICDM.2005.95, 8 pp.

Duan, Z., Tang, L., Gong, X., Zhu, Y., 2018. Personalized service recommendations for travel using trajectory pattern discovery. Int. J. Distrib. Sens. Netw. 14, 155014771876784. http://dx.doi.org/10.1177/1550147718767845.

Enge, P., 1994. The global positioning system: Signals, measurements, and performance. Int. J. Wirel. Inf. Netw. 1, 83–105. http://dx.doi.org/10.1007/BF02106512.

Ester, M., Kriegel, H.P., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. KDD '96, AAAI Press, pp. 226–231.

Galán, S.F., 2019. Comparative evaluation of region query strategies for DBSCAN clustering. Inform. Sci. 502, 76–90. http://dx.doi.org/10.1016/j.ins.2019.06.036, URL: https://www.sciencedirect.com/science/article/pii/S0020025519305742.

Giannotti, F., Nanni, M., Pinelli, F., Pedreschi, D., 2007. Trajectory pattern mining. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 330–339. http://dx.doi.org/10.1145/1281192.1281230.

Hartigan, J.A., Wong, M.A., 1979. A k-means clustering algorithm. JSTOR Appl. Stat. 28 (1), 100–108.

Hubbard, P., 1993. Interactive collision detection. In: Proceedings of IEEE Symposium on Research Frontiers in Virtual Reality. pp. 24–31. http://dx.doi.org/10.1109/VRAIS.1993.378267.

Jiménez, P., Thomas, F., Torras, C., 2001. 3D collision detection: a survey. Comput. Graph. 25 (2), 269–285.

Kockara, S., Halic, T., Bayrak, C., Iqbal, K., Rowe, R., 2009. Contact detection algorithms. J. Comput. 4, http://dx.doi.org/10.4304/jcp.4.10.1053-1063.

Kockara, S., Halic, T., Iqbal, K., Bayrak, C., Rowe, R., 2007. Collision detection: A survey. In: 2007 IEEE International Conference on Systems, Man and Cybernetics. pp. 4046–4051. http://dx.doi.org/10.1109/ICSMC.2007.4414258.

Kopp, C., May, M., Wrobel, S., 2012. Spatiotemporal modeling and analysis—Introduction and overview. Künstl. Intell. 26, http://dx.doi.org/10.1007/s13218-012-0215-2.

Li, Z., Ding, B., Han, J., Kays, R., Nye, P., 2010. Mining periodic behaviors for moving objects. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1099–1108. http://dx.doi.org/10.1145/1835804.1835942.

Lin, M., Gottschalk, S., 1998. Collision detection between geometric models: A survey. In: IMA Conf. on Mathematics of Surfaces, vol. 8.

Liu, K., Jin, X., Cheng, S., Gao, S., Yin, L., Lu, F., 2024. Act2Loc: a synthetic trajectory generation method by combining machine learning and mechanistic models. Int. J. Geogr. Inf. Sci. 38 (3), 407–431.

Liu, J., Meng, B., Shi, C., 2023a. A multi-activity view of intra-urban travel networks: A case study of Beijing. Cities 143, 104634. http://dx.doi.org/10.1016/j.cities.2023.104634, URL: https://www.sciencedirect.com/science/article/pii/S0264275123004468.

Liu, Y., Wang, X., Song, C., Chen, J., Shu, H., Wu, M., Guo, S., Huang, Q., Pei, T., 2023b. Quantifying human mobility resilience to the COVID-19 pandemic: A case study of Beijing, China. Sustainable Cities Soc. 89, 104314. http://dx.doi.org/10.1016/j.scs.2022.104314, URL: https://www.sciencedirect.com/science/article/pii/S2210670722006187.

Ma, Y., Zhu, X., Zhang, S., Yang, R., Wang, W., Manocha, D., 2019. TrafficPredict: Trajectory prediction for heterogeneous traffic-agents. Proc. AAAI Conf. Artif. Intell. 33, 6120–6127. http://dx.doi.org/10.1609/aaai.v33i01.33016120.

Majeed, A., Hwang, S.O., 2021. A comprehensive analysis of privacy protection techniques developed for COVID-19 pandemic. IEEE Access 9, 164159–164187. http://dx.doi.org/10.1109/ACCESS.2021.3130610.

Mazimpaka, J.D., 2016. Trajectory data mining: A review of methods and applications. J. Spatial Inf. Sci. 13, http://dx.doi.org/10.5311/JOSIS.2016.13.263.

Miltenberger, A., Pfahl, S., Wernli, H., 2013. An online trajectory module (version 1.0) for the nonhydrostatic numerical weather prediction model COSMO. Geosci. Model Dev. 6, http://dx.doi.org/10.5194/gmd-6-1989-2013.

Moreno, F., Pineda, A., Fileto, R., Bogorny, V., 2014. SMOT+: Extending the SMOT algorithm for discovering stops in nested sites. Comput. Inform. 33, 327–342.

Nievergelt, J., 2000. Exhaustive search, combinatorial optimization and enumeration: Exploring the potential of raw computing power. pp. 87–125. http://dx.doi.org/10.1007/3-540-44411-4_2, vol. 1963.

Palma, A., Bogorny, V., Kuijpers, B., Alvares, L., 2008. A clustering-based approach for discovering interesting places in trajectories. In: SAC. pp. 863–868. http://dx.doi.org/10.1145/1363686.1363886.

Patel, D., Sheng, C., Hsu, W., Lee, M., 2012. Incorporating duration information for trajectory classification. pp. 1132–1143. http://dx.doi.org/10.1109/ICDE.2012.72.

Qu, B., Yang, W., Cui, G., Wang, X., 2019. Profitable taxi travel route recommendation based on big taxi trajectory data. IEEE Trans. Intell. Transp. Syst. PP, 1–16. http://dx.doi.org/10.1109/TITS.2019.2897776.

Shamolin, M., 2020. Solution of the diagnostic problem in the cases of precise and inaccurate trajectory measurements. J. Math. Sci. 250, 942–963. http://dx.doi.org/10.1007/s10958-020-05056-w.

Trajcevski, G., 2011. Uncertainty in spatial trajectories. pp. 63–107. http://dx.doi.org/10.1007/978-1-4614-1629-6_3.

Šveda, M., Madajová, M.S., 2023. Estimating distance decay of intra-urban trips using mobile phone data: The case of Bratislava, Slovakia. J. Transp. Geogr. 107, 103552. http://dx.doi.org/10.1016/j.jtrangeo.2023.103552, URL: https://www.sciencedirect.com/science/article/pii/S0966692323000248.

Xing, X., Yuan, Y., Huang, Z., Peng, X., Zhao, P., Liu, Y., 2022. Flow trace: A novel representation of intra-urban movement dynamics. Comput. Environ. Urban Syst. 96, 101832. http://dx.doi.org/10.1016/j.compenvurbsys.2022.101832, URL: https://www.sciencedirect.com/science/article/pii/S019897152200076X.

Yang, W., Zhao, Y., Zheng, B., Liu, G., Zheng, K., 2018. Modeling travel behavior similarity with trajectory embedding. pp. 630–646. http://dx.doi.org/10.1007/978-3-319-91452-7_41.

Yin, L., Lin, N., Zhao, Z., 2021. Mining daily activity chains from large-scale mobile phone location data. Cities 109, 103013. http://dx.doi.org/10.1016/j.cities.2020.103013, URL: https://www.sciencedirect.com/science/article/pii/S0264275120313615.

Zhang, D., Lee, K., Lee, I., 2019. Semantic periodic pattern mining from spatio-temporal trajectories. Inform. Sci. 502, 164–189. http://dx.doi.org/10.1016/j.ins.2019.06.035, URL: https://www.sciencedirect.com/science/article/pii/S0020025519305729.

Zhang, T., Ramakrishnan, R., Livny, M., 1996. BIRCH: an efficient data clustering method for very large databases. In: SIGMOD. pp. 103–114.

Zheng, Y., 2015. Trajectory data mining: An overview. ACM Trans. Intell. Syst. Technol. 6, 1–41. http://dx.doi.org/10.1145/2743025.

Zheng, Y., Xie, X., Ma, W.-Y., 2010. GeoLife: A collaborative social networking service among user, location and trajectory. IEEE Data(base) Eng. Bull.